Congestion Control for Mixed-Loss Environments

Robert C. Durst
The MITRE Corporation
+1 703 883-7535
durst@mitre.org
www.scps.org/scps

Packet Loss and Delay Bound TCP's Performance

- TCP assumes that all loss is congestion,
  - cuts its transmission rate in half any time that data loss occurs during a round trip time
  - transmission rate rebuilds slowly
- Mathis, et al show the effects of loss on TCP's Congestion Avoidance algorithm:

$$BW < \left(\frac{MSS}{RTT}\right)\frac{1}{\sqrt{p}}$$

BW = bandwidth

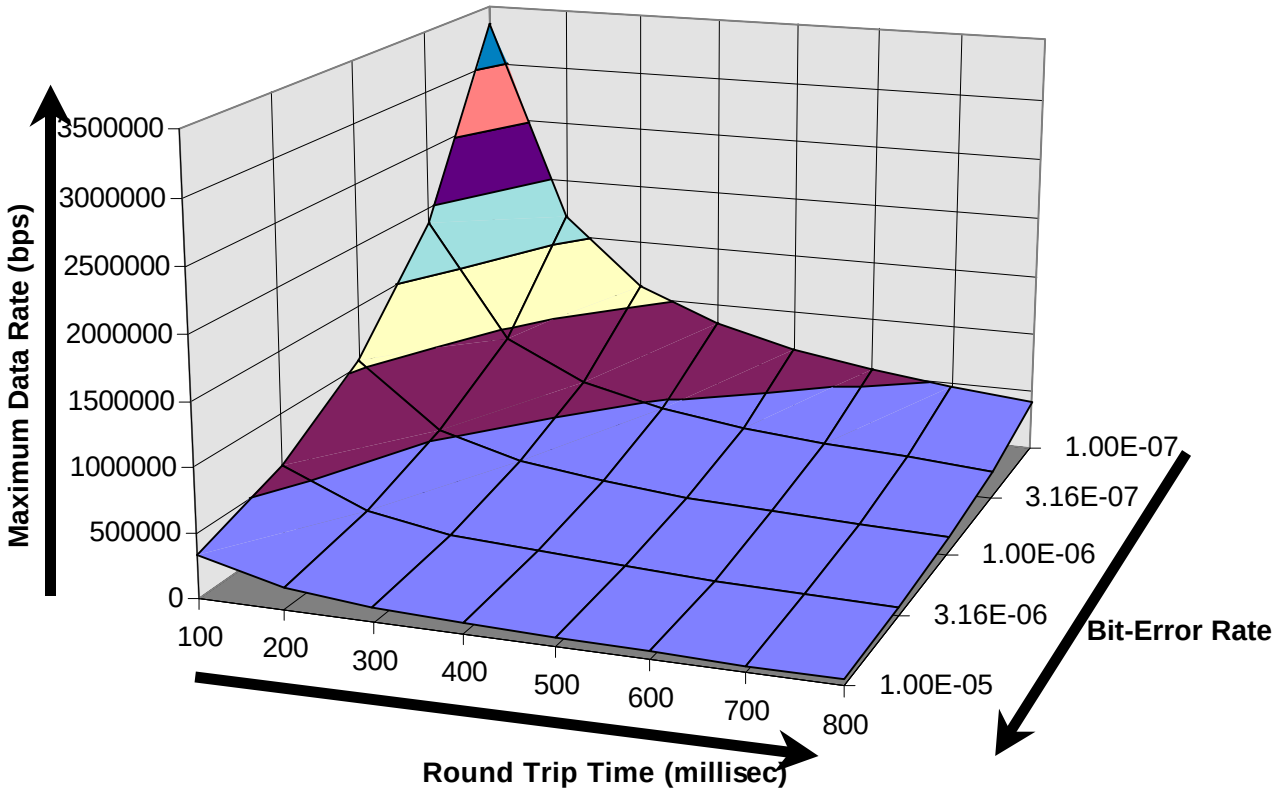
MSS = maximum TCP segment size

RTT = Round Trip Time

p = packet loss probability

- When packet loss is not due to congestion, e.g., due to error or channel outage, the result is a limit on TCP's rate that is independent of channel rate (too low)

Loss and Delay Limit TCP's Maximum Sustainable Rate



Notes:
Packet size = 1500-bytes (1448-byte TCP MSS)
Packet errors modeled according to Bernoulli Sequence

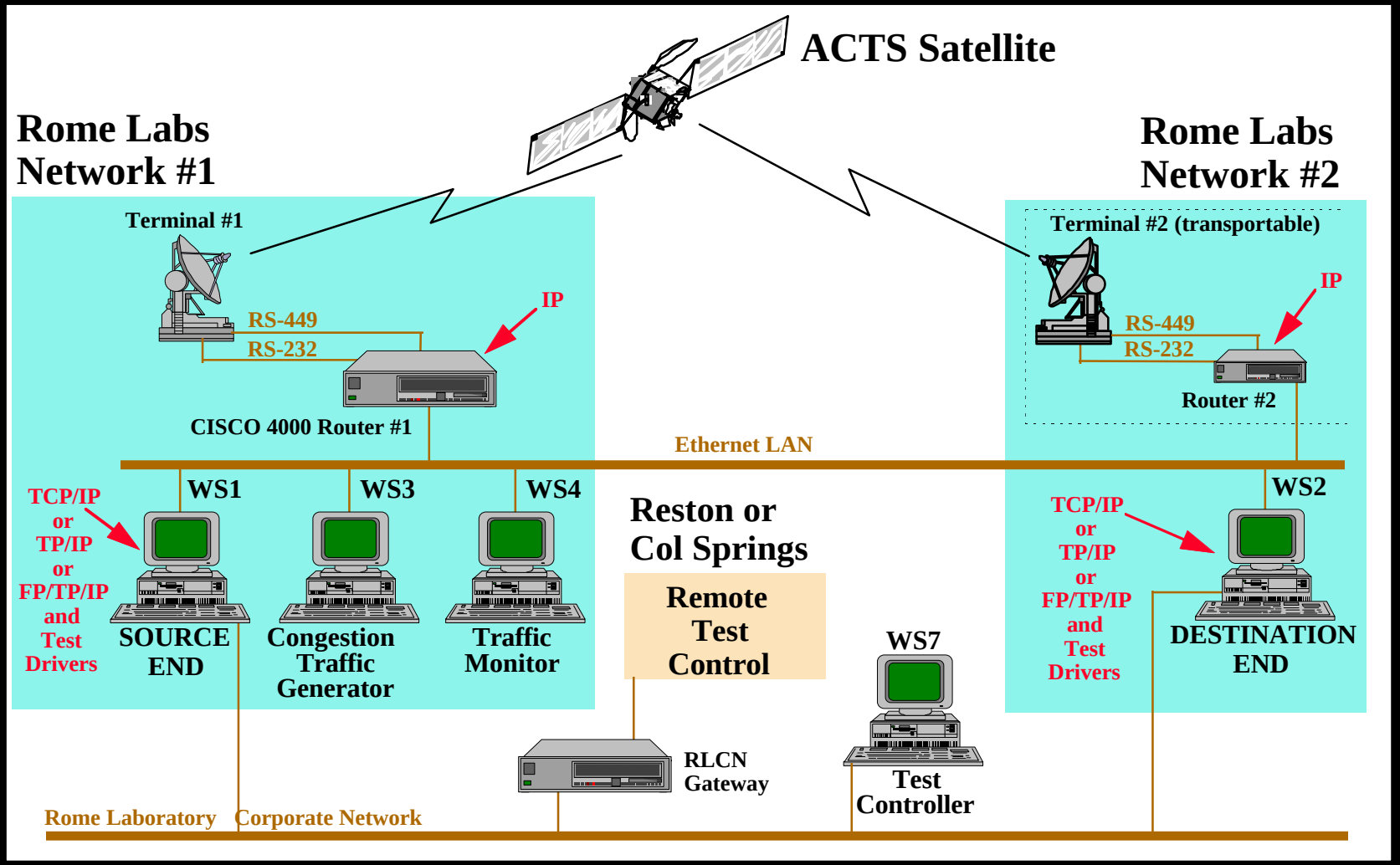
Our goal: Effective Congestion Control in the Presence of Errors

- Much of our work relates to the satellite environment, where RTTs are large (so recovery is slow) and links may be error-prone
- As higher-rate RF-based media become integrated with the Internet, effects of *non-congestion loss* will become more pronounced
- We are attempting to develop congestion control mechanisms that are effective, fair, and that do not use loss as an indication of congestion
- Similar and related work
  - TCP-Vegas - measures changes in throughput and infers the amount of queuing in the network
  - Packet-Pair - Attempts to determine bottleneck data rate by examining packet spacing

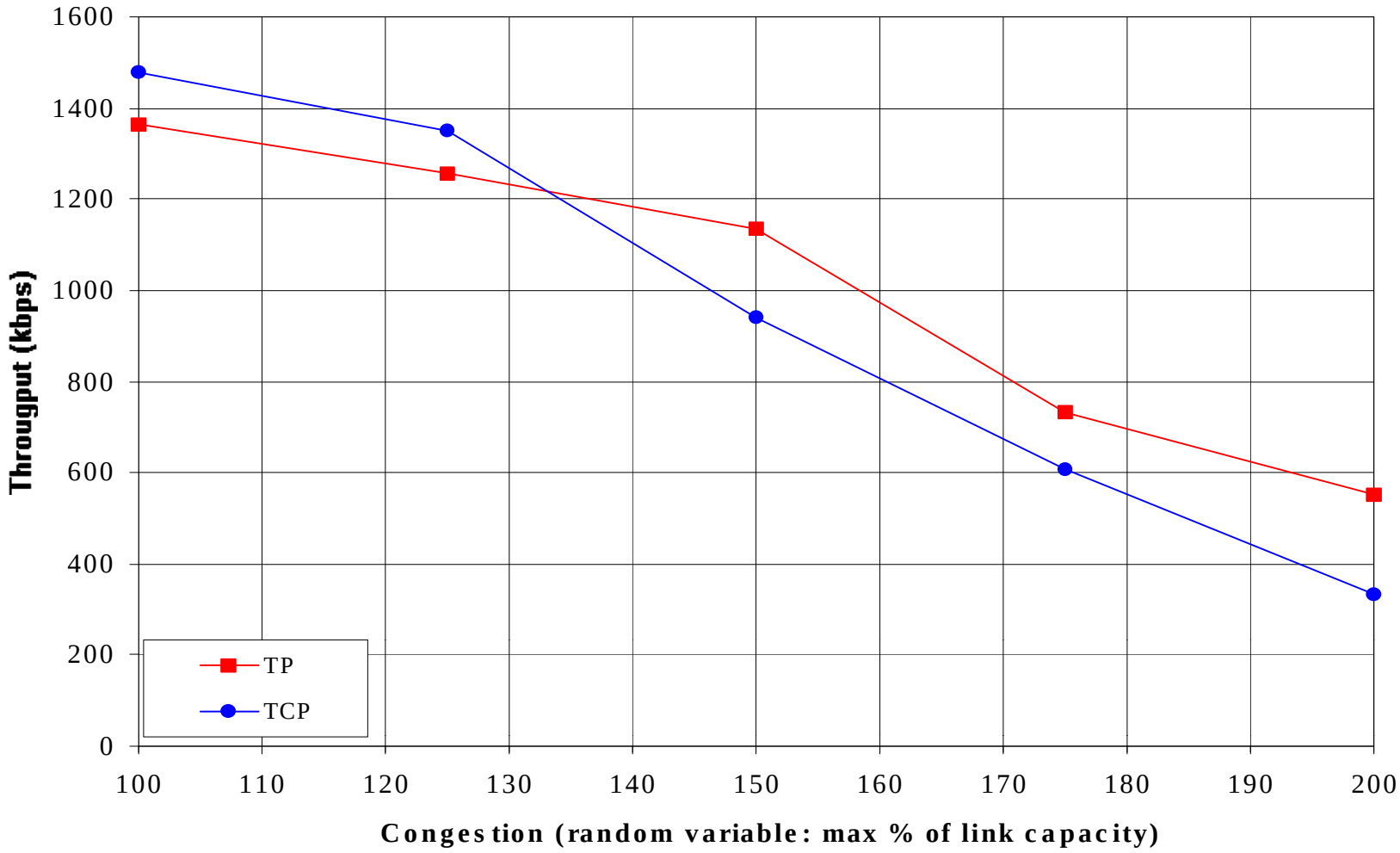
TCP-Vegas and Modifications

- We used TCP-Vegas as the basis for extension
  - Two modes of operation:
    - Exponential - similar to TCP's slow start, but slower
    - Linear - analogous to TCP's congestion avoidance
  - In it's linear mode, Vegas can increase or decrease congestion window by 1 packet/RTT or leave it unchanged (TCP can only increase)
  - Still reduces congestion window when loss is detected
- Our initial modifications:
  - Do not reduce congestion window upon loss
  - Add Explicit Congestion Notification from the network (to avoid problems with starting up in a congested environment)
  - Integrated rate control with congestion control

Comparative Results: Test Configuration

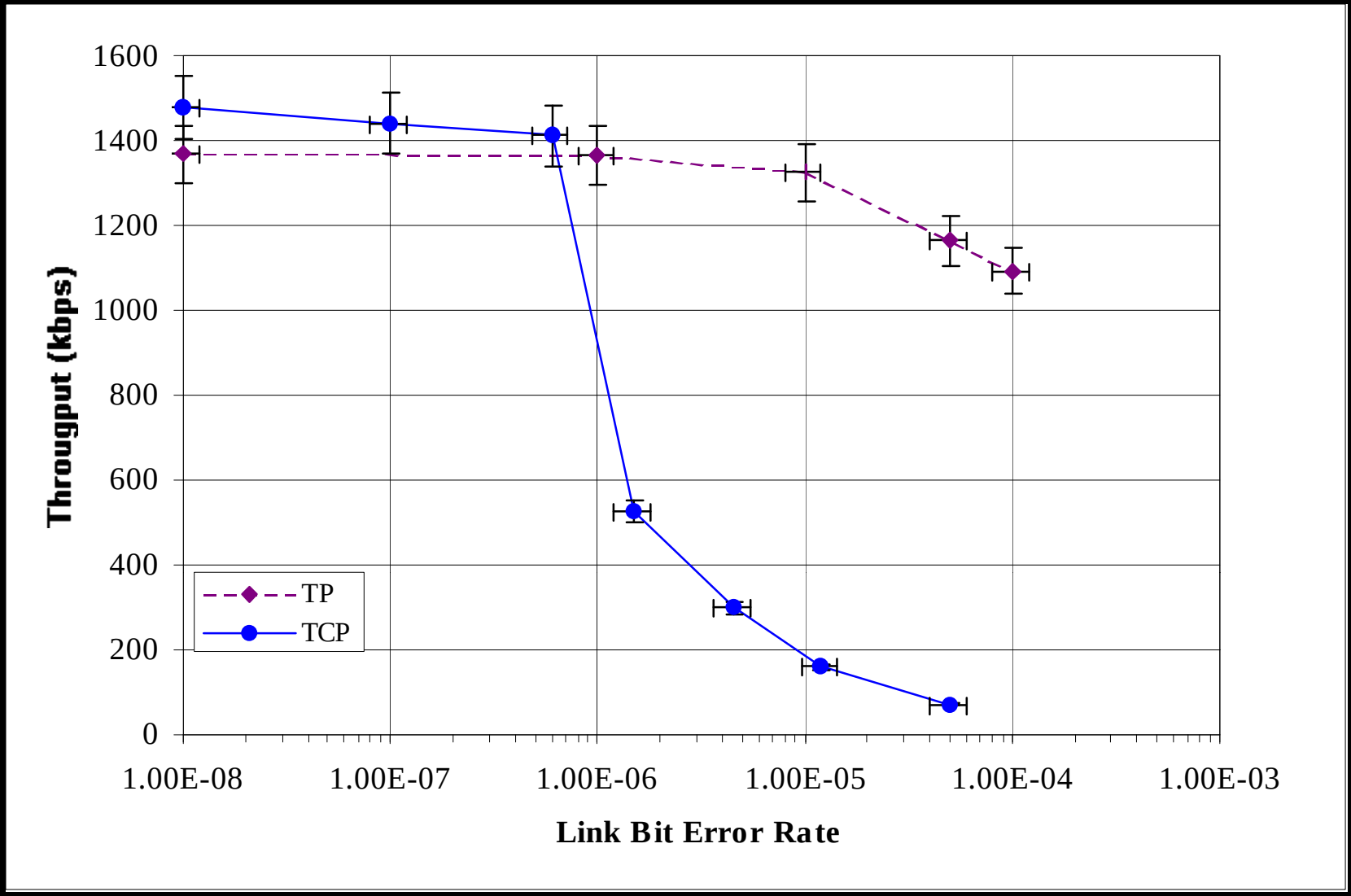


Comparative Congestion Response



MITRE

Comparative Corruption Response



MITRE

Problems with this approach

- **TCP-Vegas's slow start is too slow (doubles the transmission rate every *other* round trip time, but still can overrun available bandwidth)**
- **With (very) high bandwidth-delay product environments, need to reduce congestion window more quickly than 1 packet/RTT**
- **Uses round-trip times to calculate *data* throughput -- "noise" on the ACK channel corrupts calculation**
- **Becomes completely confused by large changes in propagation delays**

Alternatives

- Fix our modified Vegas
  - Use a different slow start algorithm
  - Revise algorithm to reduce congestion window proportional to queueing detected
  - Synchronize sender and receiver clocks to eliminate ACK channel timing effects
  - Still doesn't handle changing propagation delays
- Use a Receiver-Based Packet-Pair or Packet Bunch Mode-based approach to measure available bandwidth
  - Still must synchronize clocks
  - May want to integrate with TCP-Vegas to "damp" changes

Conclusions

- **We believe that it is possible to develop a congestion control mechanism that works effectively *but does not depend on loss as an indication of congestion***
- **Our revisions to TCP-Vegas are roughly comparable to TCP's congestion control response in the presence of congestion, and superior to TCP's in the presence of corruption**
  - **Problems remain with the Vegas-based approach**
  - **May require use of an alternative capacity measurement mechanism, such as Packet-Pair or Packet Bunch Mode**
  - **These approaches require synchronization of sender and receiver clocks to allow focus on *data* channel**
- **Deployment will be challenging**